AFRL-SR-AR-TR-08-0184

REPORT DOCUMENTATION PAGE

1. REPORT DATE *(DD-MM-YYYY)*	2. REPORT TYPE	3. DATES COVERED *(From - To)*
15-02-2008	Final Performance Report	April 2007 - January 2008

4. TITLE AND SUBTITLE
Dynamical Characteristics of Hierarchical Hybrid System for Multiple Satellite Control

5a. CONTRACT NUMBER

5b. GRANT NUMBER
FA9550-07-1-0319

5c. PROGRAM ELEMENT NUMBER

6. AUTHOR(S)
Won, Chang-Hee

5d. PROJECT NUMBER

5e. TASK NUMBER

5f. WORK UNIT NUMBER

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
Temple University
1801 N. Broad Street, Philadelphia, PA 19122-6003

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)
USAF, AFRL, AF Office of Scientific Research, 875 N. Randolph St. Room 3112, Arlington, VA 22203
OFC of Naval RSCH, Chicago Regional Office, 230 South Dearborn, Room 380, Chicago, IL 60605-1595
NL

10. SPONSOR/MONITOR'S ACRONYM(S)
AFOSR/ONRRO

11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION/AVAILABILITY STATEMENT
Distribution A: Approved for Public Release

13. SUPPLEMENTARY NOTES

14. ABSTRACT
The objective of this proposal is to investigate the dynamical properties of a hierarchical hybrid system. In the prior studies, we have developed a three-tier, hybrid system architecture for multiple remote sensing satellite control. There, we concentrated in modeling and analysis of a dynamical system whose solution only depends on the initial states. In this study, we propose to introduce another hybrid automaton with input variables. Then we will study reachability, safety, and stability properties of the hybrid system. This study will lead to a software verification tool. We will utilize game theory for automata and continuous dynamical systems. For the low-level control, we will investigate the use of statistical control and game theory for N-players. This will lead to a more effective low-level controller.

15. SUBJECT TERMS
Hierarchical hybrid system, multiple satellite control, reachability

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT	b. ABSTRACT	c. THIS PAGE		30	Chang-Hee Won
					19b. TELEPHONE NUMBER *(Include area code)* (215) 204-6158

Final Report

Dynamical Characteristics of Hierarchical Hybrid System for Multiple Satellite Control

Grant Award Number FA9550-07-1-0319

Submitted to Air Force Research Laboratory

Chang-Hee Won

CSNAP Laboratory

Department of Electrical and Computer Engineering

Temple University

6 February 2008

Abstract

We have developed a hierarchical hybrid system architecture with inputs for multiple satellite control. We re-developed the hierarchical hybrid automaton of all three tiers incorporating the input variables. Then we investigated the dynamical properties of the hybrid system. We defined nonblocking, backward reachability, forward reachability, equilibrium, and stability properties. We used backward reachability concept to solve an autonomous collision avoidance application. In collision avoidance problem, the satellites autonomously determine the possibility of the collision and change the orbit in order to avoid collision. We have developed a Matlab code for coplanar collision avoidance problem. The solution was obtained using the backward reachability concept and game theory. The game problem was solved using the level set method. On the low level control, we have developed statistical game control theory. For the two player stochastic game problem, we developed the optimal solution. We applied this statistical game idea to a satellite attitude control. We used a commercial satellite attitude model developed by the PI in the past, and applied statistical control theory to control the attitude of the satellite. We found out that this idea generalizes stochastic H-infinity controller. We obtained performance and stability tradeoff with an extra degree of freedom using statistical game theory. We presented these results in American Control Conference; and Control and Decision Conference.

Hierarchical Hybrid System Model

Hybrid Automaton

A hybrid automaton H can be described mathematically as,

$$H = (Q, X, \Sigma_{in}, \Sigma_{out}, U, f, Init, Inv, E, G, R)$$

where

Q Finite set of discrete states of H ;

X Finite set of continuous variables;

Σ_{in} Finite set of discrete input variables;

Σ_{out} Finite set of discrete output variables;

U Set of the continuous input variables;

$f : Q \times X \times \Sigma_{in} \times U \to TX$ Vector field, defining the continuous flow in discrete state;

$Init \subseteq Q \times X$ Set of initial states;

$Inv : Q \to P(X)$ Invariant sets;

$E \subseteq Q \times Q$ Set of edges;

$G : E \to P(X) \times U \times \Sigma_{in}$ Guard condition;

$R : E \times X \to P(X)$ Reset map.

The set Q is simply the set of discrete states where the system is allowed to exist. The set X will represent all of the continuous variables that are possible in each of the states of Q. The vector field, *f,* usually consists of time dependent functions such as differential equations that describe how each of the variables in X changes over time. TX denotes the tangent bundle of X. The initial states are simply the initial values of the continuous variables. The invariant set, *Inv,* denotes the set where each continuous variable belongs to as long as the discrete state remains in $q \in Q$. The invariant set is also called the domain. $P(X)$ denotes the set of all subsets of X. Σ_{in} and Σ_{out} denote the set of all

discrete inputs and discrete outputs. U is the set of all continuous input variables which includes both controllable inputs and uncontrollable inputs (disturbances). The set of edges, E, describe what transitions are allowed to occur between states. The guard conditions, G, describe the events that must occur for a transition to take place. Finally, the reset map, R, is the set of conditions that cause the system to enter its initial state.

We can model Ground Control Station automaton using discrete event system modelling, there are four discrete states q_1 , q_2 , q_3 , and q_4 , therefore $Q = \{q_1 \quad q_2 \quad q_3 \quad q_4\}$. Where

- q_1 : Idle State. In this state, GCS waits for the command parameters including the mission start time, mission altitude, longitude and latitude of the target area. If GCS receives the command parameters, it will generate the GCS_Command and output this command to Mother-ship and then system will switch from q_1 to q_2 , Mission_Schedule_Status_Check State. If GCS receives a ranging request from a target satellite, then the system will do the ranging to the target satellite, send the ranging correction command and jump from Idle state q_1 to Ranging_Correction State q_4 .
- q_2 : Mission_Schedule_Status_Check State. In this state, the ground control station waits for the mission schedule status message from the Mother-ships. If at least one Mother-ship sends back scheduling success message via the MSi_Schedule#_Finish event, that means the Mother-ships i have parsed the GCS_Command and distribute the mission to the proper clusters. The system will then jump to q_3 ,

Task_Status_Check State, if the tags MSi_Schedule#_Finish = 1 are received for all i, showing that the command has been successfully scheduled. The "#" is going to represent the command number so that there can be more than one command in the operation. If system receives a Ranging_Request from a target satellite (can be either Mother-ship or Agent), the system will jump from q_2 to q_4, Ranging_Correction State, and sends the ranging correction information to the satellite which sent the request.

- q_3: Task_Status_Check State. In this state, the system will wait for the feedback from Mother-ships indicating the status of the mission. If the scheduled mission has been successfully performed by some clusters, Mother-ships of the corresponding clusters will send a feedback message, which is called Task#_Status_OK tag, to GCS. The "#" will be the mission number. If Task#_Status_OK = 1, it means that the mission was accomplished so that the system will jump back to Idle state q_1 and waiting for the new mission command. If the Task#_Status_OK = 0, it means the mission number, #, failed during the last operation. The operator may send another GCS_Command at this point and jump to Mission_Schedule_Status_Check State, q_2 to let the Mother-ships reschedule the mission. If in state q_3, GCS receives a ranging request from a target satellite, then the system will do the ranging to the target satellite, send the ranging correction command and jump to Ranging_Correction state q_4.
- q_4: Ranging_Correction State. In this state, GCS waits for the feedback message from the target (Mother-ship or Agent) which requests a ranging correction. If the

target sends back a message that Ranging_Status_OK =1, which means that the target successfully corrects its altitude error, then, the system will jump back to the previous state where it transited from. If Ranging_Status_OK =0, the system will stay in Ranging_Correction State until a message Ranging_Status_OK =1 shows up or a Time_Out event comes out. In the later case, GCS will do the ranging again and resend the ranging correction message to the target. When GCS in state q_4 receives the Schedule#_Finish messages or Task#_Status_OK messages, the system will save the value of the messages, therefore, as soon as the system jump back to the previous states, it can recall these values and perform the corresponding operations.

The discrete events are grouped into input set and output set denoted by Σ_{in} and Σ_{out}, where

$$\Sigma_{in} = \{\varepsilon_{in1}, \varepsilon_{in2}, \varepsilon_{in3}, \varepsilon_{in4}, \varepsilon_{in5}\} = \begin{Bmatrix} Command\_Paremeters,\ Task\#\_Status\_OK,\ Schedule\#\_Finish, \\ Ranging\_Status\_OK,\ Ranging\_Request \end{Bmatrix};$$

$$\Sigma_{out} = \{\varepsilon_{o1}, \varepsilon_{o2}\} = \{GCS\_Command,\ Ranging\_Correction\}.$$

The Invariant set of $Q = \{q_1\ \ q_2\ \ q_3\ \ q_4\}$ are given as follows,

$$Inv(q_1) = \{(Command\_Parameters = 0) \wedge (Ranging\_Request = 0)\},$$

$$Inv(q_2) = \{(Schedule\#\_Complete = 0) \wedge (Ranging\_Request = 0)\},$$

$$Inv(q_3) = \{(Task\#\_Status\_OK = 0) \wedge (Ranging\_Request = 0)\},$$

$$Inv(q_4) = \{(Ranging\_Status\_OK = 0)\}.$$

The discrete transitions of the system are:

$$E = \left\{ \begin{bmatrix} (q_1, q_2) & (q_2, q_4) & (q_4, q_2) & (q_2, q_3) & (q_3, q_2) \\ (q_3, q_4) & (q_4, q_3) & (q_3, q_1) & (q_1, q_4) & (q_4, q_1) \end{bmatrix} \right\}.$$

The guard conditions are given as:

$$\begin{bmatrix} G_1 \\ G_2 \\ G_3 \\ G_4 \\ G_5 \\ G_6 \\ G_7 \\ G_8 \\ G_9 \\ G_{10} \end{bmatrix} = \begin{Bmatrix} (q_1, q_2) \Rightarrow \{[(Command\_Parameters \neq \varnothing)]\} \\ (q_2, q_4) \Rightarrow \{[(Ranging\_Request = 1)]\} \\ (q_4, q_2) \Rightarrow \{[(Ranging\_Status\_OK = 1) \vee (Ranging\_Timeout = 1)]\} \\ (q_2, q_3) \Rightarrow \{[(Schedule\#\_Finish = 1)]\} \\ (q_3, q_2) \Rightarrow \{[(Task\#\_Status\_OK = 0)]\} \\ (q_3, q_4) \Rightarrow \{[(Ranging\_Request = 1)]\} \\ (q_4, q_3) \Rightarrow \{[(Ranging\_Status\_OK = 1) \vee (Ranging\_Timeout = 1)]\} \\ (q_3, q_1) \Rightarrow \{[(Task\#\_Status\_OK = 1)]\} \\ (q_1, q_4) \Rightarrow \{[(Ranging\_Request = 1)]\} \\ (q_4, q_1) \Rightarrow \{[(Ranging\_Status\_OK = 1) \vee (Ranging\_Timeout = 1)]\} \end{Bmatrix}.$$

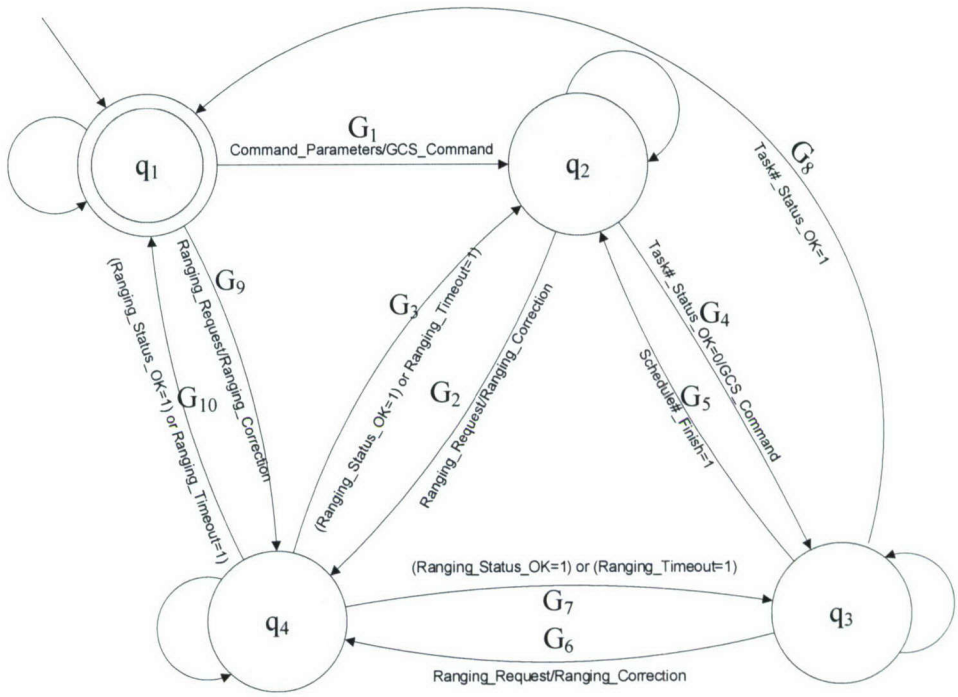

Figure 1. State Transition Diagram for Ground Control Station

The Mother-ship functions are given as follows.

1) Control the Agents and Mother-ship itself in a cluster to implement orbit and attitude change.
2) Receive the mission command from GCS, parse the command for the task parameters calculation including orbit parameters and attitude for each Agent and MS itself.
3) Communicate with other MSs to distribute the command from GCS and coordinate the operation of the different clusters.
4) Schedule the tasks for each Agent in the cluster.

5) If all the Agents in cluster cannot schedule the task, MS will send message to other MSs so that other MSs can resume the task, at the same time MS will send a mission status message back to the GCS.

Figure 2 shows the state transition diagram for the Mother-ship. The functions are divided into two subsystems. First subsystem is dealing with command generating and task scheduling, which consists of three states, q_1, q_2, and q_3, and the second subsystem is MS orbit and attitude control system which controls the orbit and attitude of MS itself, which includes three states, q_4, q_5, and q_6, where

- q_1 : Idle State. In this state, MS will monitor the command from GCS, if GCS_Command $\neq \varnothing$, the system will go to q_2, Task_Scheduling State.
- q_2: Task_Scheduling State. In this state, the system will analyze the GCS command parameters. If the cluster meets the requirements of the GCS_Command, MS will decompose the mission to a series of tasks for the components (Agents and Mother-ship) in the cluster by calculating the orbit and attitude parameters as well as the task start time and end time for each component. After the mission is decomposed, MS will send a tag called MS*i*_Task#_Schedule_OK to GCS to notify that Mother-ship *i* has finished the task scheduling. And then, the system will jump to state q_3, Agent _Control State.
- q_3: Agent _Control State. In this state, MS will control the orbit and attitude of the Agents to perform the scheduled tasks. For each Agent, at the scheduled task start time, MS will send the orbit and attitude command, which we call

Ag*j*_Task#_Command, to the Agent *j* to control the behavior of the Agent to perform the task. In q_3 state, MS also monitors the feedback messages from Agents. The message is the tag: Ag*j*_Task#_Finish which indicates whether the Agent *j* has performed the task successfully. If the mission needs MS itself to perform the orbit and attitude maneuver at the specific time, the system will jump from q_3 to q_4, which belongs to the subsystem named MS orbit and attitude control system at the scheduled time.

- q_4: MS_Orbit_Check State. In this state, MS checks the status of *MS_Orbit_Attitude Command* to see if there is a requirement to do the orbit and attitude change. If *MS_Orbit_Attitude Command* shows that the current orbit and attitude do not match the command specifics, the satellite will jump to the q_5 to start an orbit maneuver and attitude adjustment.

- q_5: MS_Attitude_Adjustment State. In this state, MS adjusts its attitude to the Euler angles calculated from the *MS_Orbit_Attitude Command* specifics. It will first check the current altitude with the target altitude, if the current altitude does not match target altitude, MS will adjust its attitude to thruster firing mode and jump to q_6 to perform an orbit maneuver. Otherwise, MS will check the current attitude with the calculated attitude for the task. If there is a difference between current attitude and the calculated attitude, MS will perform an attitude adjustment, and then it will jump back to q_4, MS_Orbit_Check State with a tag that MS_Orbit_Finish=1.

- q_6: Orbit_Maneuver State. In this state MS changes its altitude to the target altitude determined by the *MS_Orbit_Attitude Command.* If MS is not in the thruster firing

mode, it will jump to q_5 to change its attitude to the thruster firing mode to boost the satellite to the target altitude. If MS is already in the thruster firing mode, it will perform a thruster fire to shoot itself to the target orbit. After performing the orbit maneuver, MS will jump back to q_5 to adjust the Euler angles to reach target attitude specification.

The continuous variables only exist in the MS orbit and attitude control subsystem, that is, they do not affect states q_1, q_2 and q_3. The finite set of real valued continuous variables for MS model is defined as, $X = \{x_1, x_2, x_3, x_4, x_5, x_6, x_7\}$, where x_1 represents the altitude (h) of the satellite, x_2 represents angular velocity ω_x, x_3 represents the angular velocity $\delta\omega_y$, x_4 represents the angular velocity ω_z, x_5 represents the roll Euler angle ϕ, x_6 is the pitch Euler angle θ, x_7 is the yaw Euler angle ψ.

The controllable continuous inputs are the angular velocities of the four reaction wheels. $U = \{u_1, u_2, u_3, u_4\}$, $u_1 = \Omega_1$, $u_2 = \Omega_2$, $u_3 = \Omega_3$, $u_4 = \Omega_4$.

The continuous dynamics are specified as

$$f(q_4, x, u) = \left[\gamma \;\; \omega_x \;\; \delta\omega_y \;\; \omega_z \;\; \phi \;\; \theta \;\; \psi \;\; \Omega_1 \;\; \Omega_2 \;\; \Omega_3 \;\; \Omega_4\right]^T,$$

$f(q_5, x, u)$

$$= \begin{bmatrix} h - F \times \Delta t \\ \int \left(\phi \times 9.2334\text{e-}7 + \omega_Z \times 2.8937\text{e-}4 + \Omega_1 \times 2.1753\text{e-}8 + \Omega_2 \times 2.1753\text{e-}8 + \Omega_3 \times 2.1753\text{e-}8 + \Omega_4 \times 2.1753\text{e-}8 \right) dt \\ \int \left(-\theta \times 2.2257\text{e-}6 \right) dt \\ \int \left(-\omega_X \times 8.3747\text{e-}4 - \Omega_1 \times 3.0565\text{e-}8 + \Omega_2 \times 3.0565\text{e-}8 + \Omega_3 \times 3.0565\text{e-}8 - \Omega_4 \times 3.0565\text{e-}8 \right) dt \\ \int \left(\psi \times 1.0636\text{e-}3 + \omega_X \right) dt \\ \int \left(\delta\omega_y \right) dt \\ \int \left(-\phi \times 1.0636\text{e-}3 + \omega_Z \right) dt \\ \int \left(-\phi \times 5.3309\text{e-}7 + \theta \times 1.285\text{e-}6 + \omega_X \times 4.8352\text{e-}4 - \omega_Z \times 1.6707\text{e-}4 + \Omega_1 \times 5.0877\text{e-}9 - \Omega_2 \times 3.0205\text{e-}8 - \Omega_3 \times 3.0205\text{e-}8 + \Omega_4 \times 5.0877\text{e-}9 \right) dt \\ \int \left(\phi \times 5.3309\text{e-}7 + \theta \times 1.285\text{e-}6 + \omega_X \times 4.8352\text{e-}4 + \omega_Z \times 1.6707\text{e-}4 + \Omega_1 \times 3.0205\text{e-}8 - \Omega_2 \times 5.0877\text{e-}9 - \Omega_3 \times 5.0877\text{e-}9 + \Omega_4 \times 3.0205\text{e-}8 \right) dt \\ \int \left(\phi \times 5.3309\text{e-}7 - \theta \times 1.285\text{e-}6 + \omega_X \times 4.8352\text{e-}4 + \omega_Z \times 1.6707\text{e-}4 + \Omega_1 \times 3.0205\text{e-}8 - \Omega_2 \times 5.0877\text{e-}9 - \Omega_3 \times 5.0877\text{e-}9 + \Omega_4 \times 3.0205\text{e-}8 \right) dt \\ \int \left(-\phi \times 5.3309\text{e-}7 - \theta \times 1.285\text{e-}6 + \omega_X \times 4.8352\text{e-}4 - \omega_Z \times 1.6707\text{e-}4 + \Omega_1 \times 5.0877\text{e-}9 - \Omega_2 \times 3.0205\text{e-}8 - \Omega_3 \times 3.0205\text{e-}8 + \Omega_4 \times 5.0877\text{e-}9 \right) dt \end{bmatrix},$$

$$f(q_6, x, u) = \begin{bmatrix} \dfrac{a(1-e^2)}{1+e\cos\nu} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}^T.$$

The discrete events for MS are

$$\Sigma_{in} = \{\varepsilon_{in1}, \varepsilon_{in2}, \varepsilon_{in3}\} = \{GCS\_Command,\ MSi\_Task\#\_Schedule\_OK,\ Agi\_Task\#\_Finish\};$$

$$\Sigma_{out} = \{\varepsilon_{o1}, \varepsilon_{o2}, \varepsilon_{o3}, \varepsilon_{o4}, \varepsilon_{o5}\} = \begin{Bmatrix} GCS\_Command,\ MSi\_Task\#\_Coordinator,\ Agj\_Task\#\_Command, \\ MSi\_Task\#\_Schdule\_Finish,\ MSi\_Task\#\_Finish \end{Bmatrix}.$$

The Invariant set of $Q = \{q_1 \quad q_2 \quad q_3 \quad q_4 \quad q_5 \quad q_6 \quad q_7\}$ are given as follows,

$$Inv(q_1) = \{GCS\_Command = \varnothing\},$$

$$Inv(q_2) = \begin{Bmatrix} Wait\ until\ MSj\_Task\#\_Schedule\_OK\ tags\ are\ received \\ from\ j = 0,\ 1, \ldots,\ n,\ j \neq i, n\ is\ the\ number\ of\ clusters. \end{Bmatrix},$$

$$Inv(q_3) = \left\{ \begin{matrix} Wait\ until\ all\ Agj\_Task\#\_Finish\ tags\ are\ received \\ from\ j = 0,\ 1,...,\ m,\ m\ is\ the\ number\ of\ Agents\ in\ the\ cluster\ i. \end{matrix} \right\},$$

$$Inv(q_4) = \left\{ \begin{matrix} x \in \mathbb{R}^7 :\ (x_5 = TargetRoll, x_6 = TargetPitch, x_7 = TargetYaw) \\ \wedge t < MS\_Orbit\_Command\_Time \end{matrix} \right\},$$

$$Inv(q_5) = \left\{ x \in \mathbb{R}^7 : x_5 \neq CalculatedRoll, x_6 \neq CalculatedPitch, x_7 \neq CalculatedYaw \right\},$$

$$Inv(q_6) = \left\{ x \in \mathbb{R}^7 : x_1 \neq TargetAltitude \right\}.$$

The discrete transitions of the system are:

$$E = \left\{ \begin{bmatrix} (q_1, q_2) & (q_2, q_3) & (q_3, q_1) & (q_3, q_4) & \\ (q_4, q_3) & (q_4, q_5) & (q_5, q_4) & (q_5, q_6) & (q_6, q_5) \end{bmatrix} \right\}.$$

The guard conditions are given as

$$\begin{bmatrix} G_1 \\ G_2 \\ G_3 \\ G_4 \\ G_5 \\ G_6 \\ G_7 \\ G_8 \\ G_9 \end{bmatrix} = \left\{ \begin{matrix} (q_1, q_2) \Rightarrow \left\{ GCS\_Command \neq \varnothing \right\} \\ (q_2, q_3) \Rightarrow \left\{ MSi\_Task\#\_Schedule\_OK = 1\ for\ i = 1,\ 2,\ ...,\ n \right\} \\ (q_3, q_1) \Rightarrow \left\{ \begin{matrix} Agj\_Task\#\_Finish = 1\ for\ i = 1,\ 2,\ ...,\ m \\ \wedge MS\_Orbit\_Finish = 1 \end{matrix} \right\} \\ (q_3, q_4) \Rightarrow \left\{ x \in \mathbb{R}^7 : (t \geq MS\_Orbit\_Command\_Time) \right\} \\ (q_4, q_3) \Rightarrow \left\{ MS\_Orbit\_Finish = 1 \right\} \\ (q_4, q_5) \Rightarrow \left\{ \begin{matrix} x \in \mathbb{R}^7 : (x_5 \neq CalculatedRoll, \\ x_6 \neq CalculatedPitch, x_7 \neq CalculatedYaw) \end{matrix} \right\} \\ (q_5, q_4) \Rightarrow \left\{ \begin{matrix} x \in \mathbb{R}^7 : \begin{pmatrix} x_5 = CalculatedRoll, \\ x_6 = CalculatedPitch, x_7 = CalculatedYaw \end{pmatrix} \\ \wedge (x_1 \geq Target\_Altitude) \end{matrix} \right\} \\ (q_5, q_6) \Rightarrow \left\{ x \in \mathbb{R}^7 : (|x_1| > Target\_Altitude + 2000) \right\} \\ (q_6, q_5) \Rightarrow \left\{ x \in \mathbb{R}^7 : (|x_1| \leq Target\_Altitude + 2000) \right\} \end{matrix} \right\}.$$

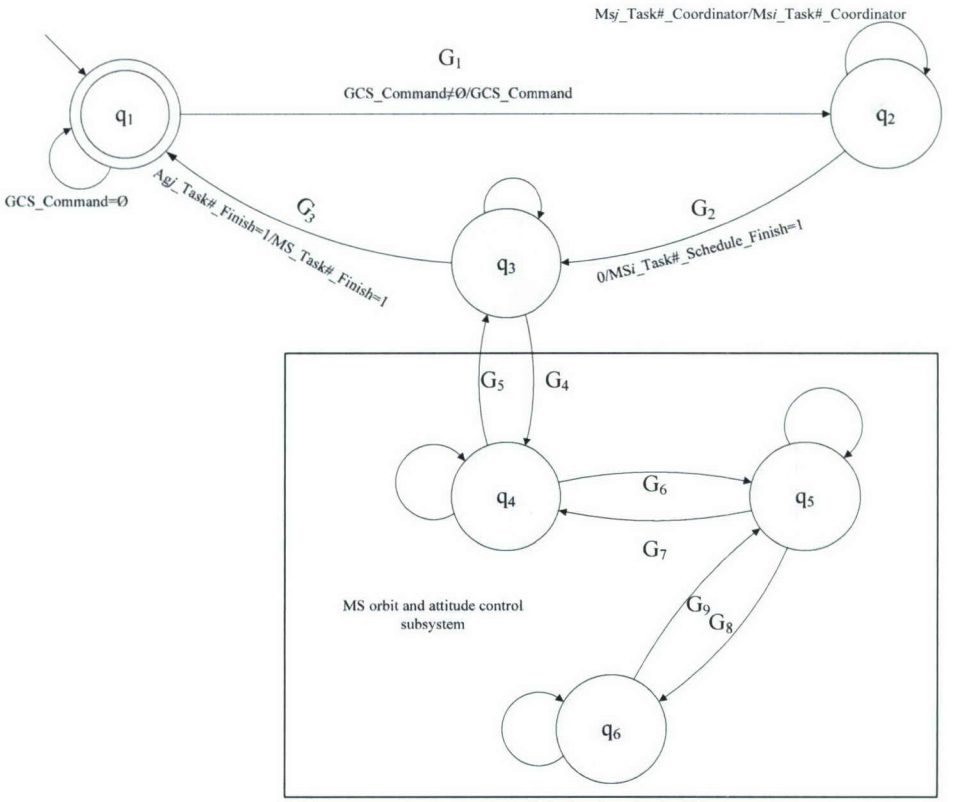

Figure 2. State Transition Diagram for the Mother-ship No.*i*

The Agent functions are as follows.

1) Receive orbit and attitude commands from MS.
2) Perform the orbit maneuver and attitude change for the scheduled tasks.
3) Send the task status feedback to MS to notify whether the task is successful.
4) Switch to MS in the case that the MS malfunctions, this operation is done autonomously by the Agents in that group via certain election.

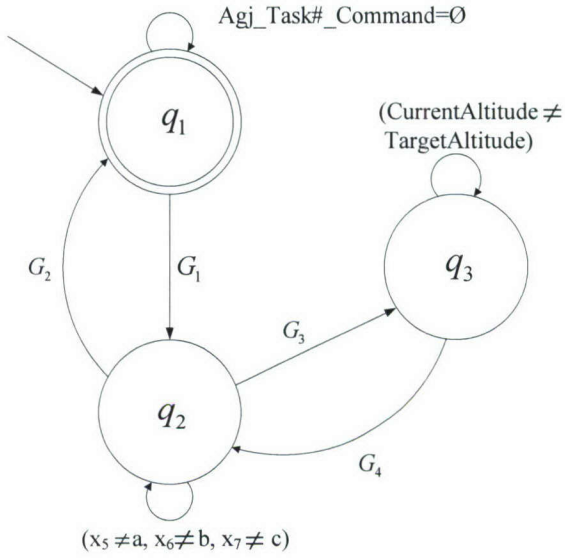

Figure 3. State transition diagram for Agent control system.

The agent control system is indeed the same as the MS orbit and attitude control subsystem in the way that they both control the satellite to perform the orbit maneuver and attitude adjustment. Therefore, Agent control system consists of three states, q_1, q_2, and q_3, where

- q_1 : Idle State. In this state, Agent will monitor the command from MS, if $Agj\_Task\#\_Command \neq \varnothing$, and the current orbit and attitude are different from the command specifics, the satellite will jump to the q_2 to start an orbit maneuver and attitude adjustment.
- q_2 : Attitude_Adjustment State. In this state, Agent will analyze the $Agj\_Task\#\_Command$ parameters including target position and target attitude. If there is a requirement for the orbit maneuver and attitude adjustment, Agent will calculate the Euler angles from $Agj\_Task\#\_Command$ parameters. Then, it will

check the current altitude with the target altitude, if the current altitude does not match target altitude, Agent will adjust its attitude to thruster firing mode and jump to q_3 to perform an orbit maneuver. If current altitude matches the target altitude, Agent will check the current attitude with the calculated attitude for the task. If there is a difference between current attitude and the calculated attitude, Agent will perform an attitude adjustment in q_2, and then jump back to q_1, Idle State, with a tag that Agent_Orbit_Finish=1.

- q_3: Orbit_ Maneuver State. In this state, Agent changes its altitude to the target altitude specified in the *Agj _Task#_Command*. If Agent is not in the thruster firing mode, it will jump to q_2 to change its attitude to the thruster firing mode to boost the satellite to the target altitude. If Agent is already in the thruster firing mode, it will perform a thruster fire to shoot itself to the target orbit. After performing the orbit maneuver, MS will jump back to q_2 to check the current attitude and the target attitude specification.

The continuous variables of the Agent control system affect states q_1, q_2 and q_3. The finite set of real valued continuous variables for Agent model is defined as, $X = \{x_1, x_2, x_3, x_4, x_5, x_6, x_7\}$, where x_1 represents the altitude (h) of the satellite, x_2 represents angular velocity ω_x, x_3 represents the angular velocity $\delta\omega_y$, x_4 represents the angular velocity ω_z, x_5 represents the roll Euler angle ϕ, x_6 is the pitch Euler angle θ, x_7 is the yaw Euler angle ψ.

The controllable continuous inputs are the angular velocities of the four reaction wheels. $U = \{u_1, u_2, u_3, u_4\}$, $u_1 = \Omega_1$, $u_2 = \Omega_2$, $u_3 = \Omega_3$, $u_4 = \Omega_4$.

The continuous dynamics are specified as

$$f(q_1, x, u) = \left[\gamma \;\; \omega_x \;\; \delta\omega_y \;\; \omega_z \;\; \phi \;\; \theta \;\; \psi \;\; \Omega_1 \;\; \Omega_2 \;\; \Omega_3 \;\; \Omega_4\right]^T,$$

$$f(q_2, x, u) = \begin{bmatrix} h - F \times \Delta t \\ \int\left(\phi \times 9.2334\text{e-}7 + \omega_Z \times 2.8937\text{e-}4 + \Omega_1 \times 2.1753\text{e-}8 + \Omega_2 \times 2.1753\text{e-}8 + \Omega_3 \times 2.1753\text{e-}8 + \Omega_4 \times 2.1753\text{e-}8\right) dt \\ \int\left(-\theta \times 2.2257\text{e-}6\right) dt \\ \int\left(-\omega_X \times 8.3747\text{e-}4 - \Omega_1 \times 3.0565\text{e-}8 + \Omega_2 \times 3.0565\text{e-}8 + \Omega_3 \times 3.0565\text{e-}8 - \Omega_4 \times 3.0565\text{e-}8\right) dt \\ \int\left(\psi \times 1.0636\text{e-}3 + \omega_X\right) dt \\ \int\left(\delta\omega_y\right) dt \\ \int\left(-\phi \times 1.0636\text{e-}3 + \omega_Z\right) dt \\ \int\left(-\phi \times 5.3309\text{e-}7 + \theta \times 1.285\text{e-}6 + \omega_X \times 4.8352\text{e-}4 - \omega_Z \times 1.6707\text{e-}4 + \Omega_1 \times 5.0877\text{e-}9 - \Omega_2 \times 3.0205\text{e-}8 - \Omega_3 \times 3.0205\text{e-}8 + \Omega_4 \times 5.0877\text{e-}9\right) dt \\ \int\left(\phi \times 5.3309\text{e-}7 + \theta \times 1.285\text{e-}6 + \omega_X \times 4.8352\text{e-}4 + \omega_Z \times 1.6707\text{e-}4 + \Omega_1 \times 3.0205\text{e-}8 - \Omega_2 \times 5.0877\text{e-}9 - \Omega_3 \times 5.0877\text{e-}9 + \Omega_4 \times 3.0205\text{e-}8\right) dt \\ \int\left(\phi \times 5.3309\text{e-}7 - \theta \times 1.285\text{e-}6 + \omega_X \times 4.8352\text{e-}4 + \omega_Z \times 1.6707\text{e-}4 + \Omega_1 \times 3.0205\text{e-}8 - \Omega_2 \times 5.0877\text{e-}9 - \Omega_3 \times 5.0877\text{e-}9 + \Omega_4 \times 3.0205\text{e-}8\right) dt \\ \int\left(-\phi \times 5.3309\text{e-}7 - \theta \times 1.285\text{e-}6 + \omega_X \times 4.8352\text{e-}4 - \omega_Z \times 1.6707\text{e-}4 + \Omega_1 \times 5.0877\text{e-}9 - \Omega_2 \times 3.0205\text{e-}8 - \Omega_3 \times 3.0205\text{e-}8 + \Omega_4 \times 5.0877\text{e-}9\right) dt \end{bmatrix},$$

$$f(q_3, x, u) = \left[\frac{a(1-e^2)}{1 + e\cos\nu} \;\; 0 \;\; 0 \;\; 0 \;\; 0 \;\; 0 \;\; 0 \;\; 0 \;\; 0 \;\; 0 \;\; 0\right]^T.$$

The discrete events for the Agent control system are

$$\Sigma_{in} = \{\varepsilon_{in1}\} = \{Agi\_Task\#\_Command\};$$

$$\Sigma_{out} = \{\varepsilon_{o1}\} = \{Agj\_Task\#\_Finish\}.$$

The Invariant set of $Q = \{q_1 \;\; q_2 \;\; q_3\}$ are given as follows,

$$Inv(q_1) = \{Agj\_Task\#\_Command = 0\},$$

$$Inv(q_2) = \left\{x \in \mathbb{R}^7 : (x_5 \neq \text{CalculatedRoll}, x_6 \neq \text{CalculatedPitch}, x_7 \neq \text{CalculatedYaw})\right\},$$

$$Inv(q_3) = Inv(q_6) = \left\{ x \in \mathbb{R}^7 : x_1 \neq TargetAltitude \right\}.$$

The discrete transitions of the system are:

$$E = \left\{ \left[(q_1, q_2) \quad (q_2, q_1) \quad (q_2, q_3) \quad (q_3, q_2) \right] \right\}.$$

The guard conditions are given as

$$\begin{bmatrix} G_1 \\ G_2 \\ G_3 \\ G_4 \end{bmatrix} = \begin{Bmatrix} (q_1, q_2) \Rightarrow \{ Agj\_Task\#\_Command \neq \varnothing \} \\ (q_2, q_1) \Rightarrow \begin{Bmatrix} x \in \mathbb{R}^7 : (x_1 = TargetAltitude) \wedge (x_5 = TaregetRoll) \\ \wedge (x_6 = TargetPitch) \wedge (x_7 = TargetYaw) \end{Bmatrix} \\ (q_2, q_3) \Rightarrow \left\{ x \in \mathbb{R}^7 : (|x_1| > Target\_Altitude + 2000) \right\} \\ (q_3, q_2) \Rightarrow \left\{ x \in \mathbb{R}^7 : (|x_1| \leq Target\_Altitude + 2000) \right\} \end{Bmatrix}.$$

Forward Reachable Set of the Satellite Hybrid Systems

Reachability is a concept of hybrid systems which states that a certain state $(q^*, x^*) \in Q \times X$ can be reached from an initial set of states through a finite execution. Formally, a state $(q^*, x^*) \in Q \times X$ is reachable if there exists a finite execution $\zeta_H(\tau, q, x, t, a, b, \delta_a, \delta_b) \in \varepsilon_H$ from the initial state (q_o, x_o) such that $\zeta_H(\tau, q, x, t^*, a, b, \delta_a, \delta_b) = (q^*, x^*)$. Where ε_H is the set of all executions of H, $\zeta_H(\tau, q_o, x_o, t^*, a, b, \delta_a, \delta_b)$ is one of the finite execution of ε_H. τ is the hybrid time trajectory, $\tau = \{[\tau_i, \tau_i']\}_{i=0}^{N}$ or $\{[\tau_i, \tau_i']\}_{i=0}^{N-1} \cup [\tau_N, \tau_N')$ with $N < \infty$. $t^* \in \tau$ is the time when the system reaches the state (q^*, x^*), a is the input, A is the set of all inputs, b is the disturbance, B is the set of all disturbances, δ_a is discrete input, $\sum_a$ is the set of all discrete inputs, δ_b is discrete disturbance, and $\sum_b$ is the set of all discrete disturbances. Then the set of reachable states can be represented as

$$Reach_H^f = \left\{ \begin{array}{l} (q^*, x^*) \in Q \times X \mid \forall b \in B, \forall \delta_b \in \sum_b, \exists a \in A, \exists \delta_a \in \sum_a, \\ \exists \zeta_H(\tau, q, x, t, a, b, \delta_a, \delta_b) \in \varepsilon_H, \text{ such that } \zeta_H(\tau, q, x, t^*, a, b, \delta_a, \delta_b) = (q^*, x^*) \end{array} \right\}.$$

Existence and Uniqueness of the Executions in a Satellite Hybrid System

To determine whether an execution of a hybrid system exists and is unique, we need to determine the non-blocking property and the deterministic property of the execution.

For the non-blocking property of the executions in the hybrid systems, we will use the definition of the forward reachable set. We define another concept, the transition set. The transition set stands for a set where the continuous evaluation is impossible. It can be expressed as

$$Trans_H = \left\{ \begin{array}{l} (q^*, x^*) \in Q \times X \mid \forall b \in B, \forall \delta_b \in \sum_b, \forall a \in A, \forall \delta_a \in \sum_a, \\ \forall \varepsilon > 0, \exists t^{**} \in [0, \varepsilon), \text{such that } \zeta_H(\tau, q\ , x, t^{**}, a, b, \delta_a, \delta_b) \notin Inv(q^*) \end{array} \right\},$$

where $\zeta_H(\tau, q\ , x, t^{**}, a, b, \delta_a, \delta_b) \in \varepsilon_H$ is an arbitrary execution which starts from (q^*, x^*) and $Inv(q^*)$ is the invariant set of the state q^*.

The non-blocking property is defined as follows. A hybrid system is non-blocking if for all $(q_o, x_o) \in Init$, at least one infinite execution $\zeta_H(\tau, q, x, t, a, b, \delta_a, \delta_b) \in \varepsilon_H$ exists, which starts from the initial state (q_o, x_o). Here the hybrid time trajectory τ is inifite.

The following lemma can be used to determine the non-blocking property of a hybrid system.

Lemma I: A hybrid system H is non-blocking if for all $(q, x) \in Reach_H \bigcap Trans_H$, there exists an infinite execution $\zeta_H(\tau, q, x, t, a, b, \delta_a, \delta_b) = (q^*, x^*)$, such that $(q,\ q^*) \in E$ and $\{x, a, b, \delta_a, \delta_b\} \in G(q, q^*)$.

Here E is the edge of the hybrid system and G is the guard condition of the transition of the hybrid system.

The deterministic property of an execution of a hybrid system is determined by the following lemma.

Lemma II: A hybrid system H is deterministic if and only if for all $(q,x) \in Reach_H$,

1) if $(x,a,b,\delta_a,\delta_b) \in G(q,q^*)$ for some $(q,q^*) \in E$, then $(q,x) \in Trans_H$;
2) if $(q,q^*) \in E$ and $(q,q^{**}) \in E$ with $q^* \neq q^{**}$, then $(x,a,b,\delta_a,\delta_b) \notin G(q,q^*) \cap G(q,q^{**})$;
3) if $(q,q^*) \in E$ and $(x,a,b,\delta_a,\delta_b) \in G(q,q^*)$, then $R(q,q^*,x)$ cotains at most one element.

Now, we present the theorem to determine the existence and uniqueness of the hybrid system, H.

Theorem I: If a hybrid system, H, satisfies Lemma I and Lemma II above, then H accepts a unique infinite execution for all $(q_o,x_o) \in Init_H$.

Backward Reachable Set of the Satellite Hybrid Systems

The backward reachable set means that the system starts from the final set of states, then follows some certain executions to go back and find the previously visited set of states. This set of states is called backward reachable set. Let the final set be the set at time t=0, then the backward reachable set $Reach_H^b$ is the set of states at time $t^* \in [-T,0]$.

$$Reach_H^b = \left\{ \begin{array}{l} (q^*,x^*) \in Q \times X, | \ \forall b \in B, \forall \delta_b \in \Sigma_b, \exists a \in A, \exists \delta_a \in \Sigma_a, \\ \exists t^* \in [-T,0], \ \zeta_H(\tau,x,q,t,a,b,\delta_a,\delta_b) \in \varepsilon_H, \\ \text{such that } \zeta_H(\tau,x,q,t^*,a,b,\delta_a,\delta_b) \in (q^*,x^*) \end{array} \right\},$$

Where a is the controllable input, b is the uncontrollable input. τ is the hybrid time trajectory, ζ_f is the executions of H which starts from the final set, q is the discrete states, x is the continuous state variable, δ_a is discrete controllable inputs and δ_b is discrete uncontrollable input.

One application of the backward reachability research is to calculate the unsafe set for the collision avoidance applications.

Satellite Collision Avoidance Example

The satellite collision avoidance model is similar to the aircraft conflict resolution model introduced by C. Tomlin and I. Mitchell etc. [1-4]. We define two types of zones for the satellite. One zone is called the unsafe zone, and the other is warning zone.

The unsafe zone is a circular area with the satellite at the center with a certain radius. The unsafe zone defines the area where the target satellite has a considerably high

probability of collision with other objects in the same area. In this model, we assume that once an object enters the unsafe zone of the satellite then the collision is inevitable. Therefore, the collision avoidance maneuver of the satellite is to control the motion of the satellite in order to keep all other objects from entering the unsafe zone.

The warning zone is an area which contains the current position of the satellite. When there is another object presenting within this zone, the satellite should take collision avoidance maneuver immediately otherwise the object will enter the unsafe zone after a certain time. The shape and the area of the warning zone depend on orbital parameters of the satellite and the incoming object and the responding time of the satellite. It should be noted that because the satellite velocity changes with respect to satellite position, the warning zone may also change according to the orbit.

In our satellite collision avoidance example, we name the satellite, which is trying to avoid the collision, as the evader, and the incoming object, which is trying to cause the collision, as the pursuer. We assume the evader does not have the knowledge about the orbit of the pursuer but evader can monitor the instantaneous pursuer position and velocity, which can be detected either by the evader itself or by the ground control station which notifies the evader. Because we cannot predict the orbit of the pursuer, we define a circular area as the warning zone. The radius of the warning zone depends on the maximum possible detected velocity of the pursuer and the responding time of the evader. Then, the evader determines the collision avoidance maneuver according to the position and the velocity when pursuer enters the warning zone.

Next we will explain how the collision avoidance maneuver is determined by the evader. The goal for the evader is to keep the pursuer from entering the unsafe zone. Therefore, we set the unsafe zone as the final set, then compute the backward reachable set which starts from the unsafe zone for a certain time interval. This interval, which ensures that the evader has enough time to respond to avoid the pursuer, is chosen as ten seconds. We shall see this ten seconds responding time is enough for the collision avoidance in the later section. Then, we choose the radius of the warning zone so that the warning zone can cover all the ten seconds backward reachable set. Then, for any pursuer which enters the warning zone, the evader has at least ten seconds to perform the collision avoidance maneuver. Because we do not know the orbital characteristics of the

pursuer, we assume that when the pursuer enters the warning zone of the evader, the velocity vectors of the pursuer and the evader do not change with time. This assumption is reasonable because when we consider the motion of a LEO/MEO/GEO object, which has an orbital period of several hours, or days, then, in a short time period, i.e. ten seconds, the change of the orbital velocity (speed and direction) is very small and can be viewed as unchanged.

Then, we need to find a way to mathematically describe the relation between evader and the pursuer. Because both the warning zone and the unsafe zone are defined with respect to the evader, we model the system using a relative coordinates with the position of evader as origin, the direction of the evader linear velocity as the x axis, the y axis is counterclockwise 90 degrees to the x axis. The position of the pursuer then can be represented in the relative coordinates x_r and y_r. The angle ψ_r represents the difference of the velocity vector from the pursuer to the evader. The radius of the unsafe circle of evader is Ru. Using these relative coordinates, the parameters and the relation between evader and the pursuer are shown in the Figure 1.

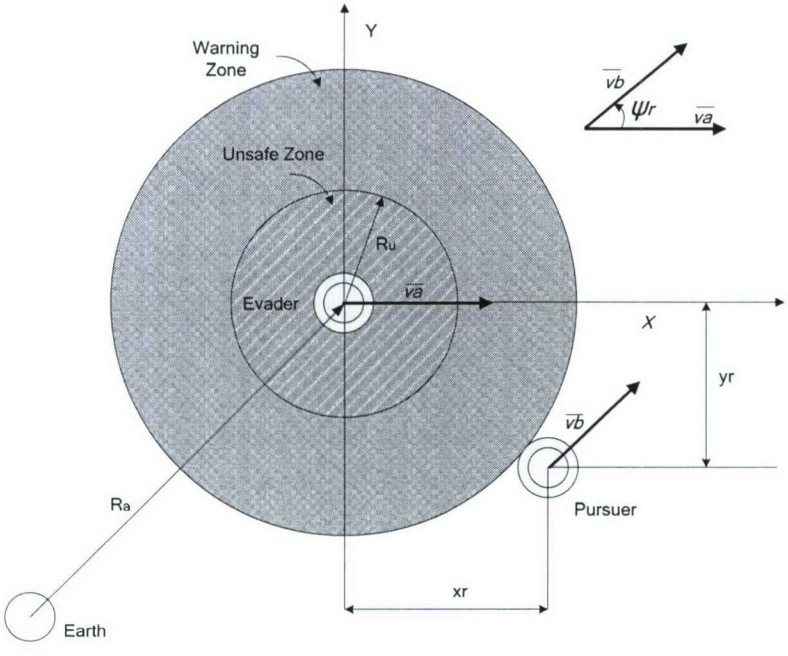

Figure 1 Evader and pursuer in relative coordinates

To simplify the problem, we make the following assumptions. The evader has a circular orbit. The pursuer and the evader inclinations are equal to each other, meaning that they are in the same orbital plane. The evader collision avoidance maneuver can be performed by changing the original orbit of the evader. In our example, we implement the orbit change by adding a Δv to the evader velocity v_a, and the Δv is the control input to the system.

The satellite dynamics in the relative coordinates can be represented by the following equations:

$$\begin{cases} \dot{x}_r = v_b \cos(\psi_r) - v_a \\ \dot{y}_r = v_b \sin(\psi_r) \end{cases}, \tag{1}$$

where

$x_r, y_r \in \mathbb{R}^2$, $\psi_r \in [0, 2\pi)$, $v_a \in V_a = [v_a - \Delta v, v_a + \Delta v]$, $v_b \in V_b = [v_{b\max}, v_{b\min}]$, $v_a, v_b \in \mathbb{R}$,

$v_a = \sqrt{\mu \frac{1}{R_a}}$, R_a is the distance from the earth center to the evader, Δv is the velocity control input, v_b is the observed pursuer instantaneous velocity which fluctuates in a range $[v_{b\max}, v_{b\min}]$, ψ_r is the angle between the pursuer velocity and evader velocity, all variables are used as scalars in (1).

The backward reachable set can be calculated using the level set method discussed in [4], we define a value function $\phi(x,t)$, which satisfies the condition $\phi(x,0) = l(x) = x_r^2 + y_r^2 - R_U^2, x \in \mathbb{R}^2$, meaning that at the final time $t=0$, the value function should represent the area of the unsafe zone. Then, the zero sublevel set of ϕ, which describers the unsafe zone of the evader, is the backwards reachable set [4]. We use $G(\tau)$ to denote the backward reachable set, and $G(\tau)$ is given by

$$G(\tau) = \{x \in \mathbb{R}^n \mid \phi(x,t) \le 0\}, \text{where t=-}\tau. \tag{2}$$

From equation (2) we notice that the backward reachable set $G(\tau)$ is determined once we find the value function $\phi(x,t)$. We use Level Set method which is discussed in detail in [4] to calculate $\phi(x,t)$.

The dynamics of the value function $\phi(x,t)$ can be presented in a Hamilton-Jacobi equation.

$$D_t\phi(x,t) = -\min\{0,\ H(x, D_x\phi(x,t))\},\ \text{for } t \in [-T,0],\ x \in \mathbb{R}^n; \tag{3}$$

Because the reachable set is non-decreasing backwards with time, therefore, in (3), we use $\min\{0,\ H(x, D_x\phi(x,t))\}$ instead of $H(x, D_x\phi(x,t))$ to ensure that $\phi(x,t_2) \le \phi(x,t_1),\ \text{for } t_1 \le t_2 \le 0.$

Let $p = D_x\phi(x,t)$, then, we find that the Hamiltonian

$$H(x,p) = \max_{v_a \in V_a} \min_{v_b \in V_b} p^T f(x, v_a, v_b). \tag{4}$$

The maxmin sign in (4) indicates that the evader control input, which is contained in v_a, always tries to maximize the Hamiltonian, while the pursuer always tries to minimize the Hamiltonian.

Combining equation (1) and (4) we have

$$\begin{aligned} H(x,p) &= \max_{v_a \in V_a} \min_{v_b \in V_b} p^T f(x, v_a, v_b) \\ &= \max_{v_a \in V_a} \min_{v_b \in V_b} \left(p_1(v_b \cos(\psi_r) - v_a) + p_2 v_b \sin(\psi_r) \right) \\ &= \max_{v_a \in V_a} \min_{v_b \in V_b} \left\{ -p_1 v_a + \left(p_1(\cos(\psi_r) + p_2 \sin(\psi_r) \right) v_b \right\}. \end{aligned} \tag{5}$$

Let switch functions $s_1 = -p_1;\ s_2 = p_1\cos(\psi_r) + p_2\sin(\psi_r)$, then, we can find the optimal v_a and v_b, that

$$v_a^* = \begin{cases} v_a + \Delta v, & \text{if } p_1 \le 0 \\ v_a + \Delta v, & \text{if } p_1 > 0 \end{cases},\ v_b^* = \begin{cases} v_{b\max}, & \text{if } p_1\cos(\psi_r) + p_2\sin(\psi_r) \le 0 \\ v_{b\min}, & \text{if } p_1\cos(\psi_r) + p_2\sin(\psi_r) > 0 \end{cases}.$$

Plugging v_a^* and v_b^* into (5), we can find the Hamiltonian. After the Hamiltonian is determined, we can use the MATLAB Level Set toolbox created by Ian Michael [4] to find the solution of the Hamilton-Jacobi equation (3), then the backward reachable set $G(\tau)$ can be determined by the equation (2).

Now we show an example to illustrate how this evader-pursuer collision avoidance model works.

Assuming evader is in the low earth orbit with an altitude 600km, then $R_a = R_{earth} + Altitude = 6371 + 600 = 6971km$, then

$\omega_a = \sqrt{\frac{\mu}{R_a^3}} = \sqrt{\frac{398600}{6971^3}} = 1.177e-6(\text{rad/s}),\ v_a = \sqrt{\mu\frac{1}{R_a}} = 7.56(\text{km/s})$, assuming $\Delta v = 2(\text{km/s})$, the velocity of pursuer is $v_b = [v_{b\max}, v_{b\min}] = [9.25(\text{km/s}), 10.25(\text{km/s})]$, the radius of the unsafe zone is $R_u = 10km$. We use the Level Set toolbox to calculate the ten seconds backward reachable set for ψ_r from $[0, 2\pi)$. The shadowed area in Figure 2 illustrates the backward reachable set of the evader.

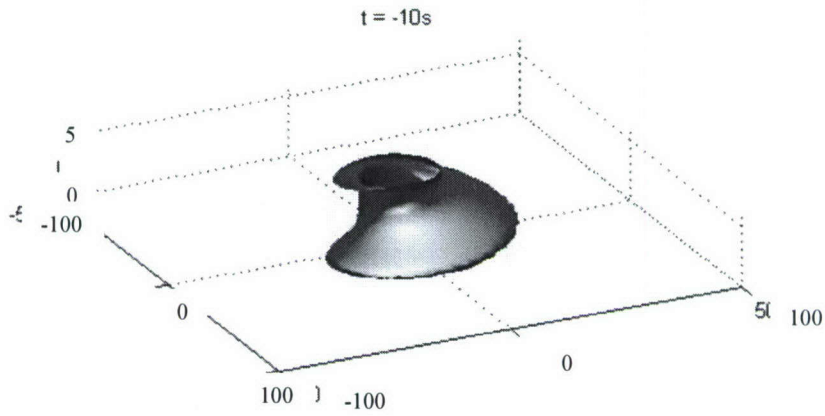

Figure 2. Backward reachable set of the evader for ψ_r from $[0, 2\pi)$

Then we will explain how the collision avoidance mechanism takes effect using the backward reachable set knowledge through a specific example. Assuming that the angle ψ_r, at the time when pursuer enters the ten seconds backward reachable set, is $\psi_r = \frac{\pi}{3}$. We pick up the hatch envelop at $\psi_r = \frac{\pi}{3}$ in Figure 2. The hatch envelop shows the ten seconds backward reachable set of the evader for $\psi_r = \frac{\pi}{3}$. The shadowed area in Figure 3 is the final unsafe set of the evader at the ending time *t=0*s. The shadowed area in Figure 4 shows the backward reachable set at *t=-10*s (because we start from *t=0*s and

compute the backward reachable set, therefore the ten second reachable set is actually at t=-*10*s).

From Figure 4, we know that if the pursuer is in the reachable set of the evader, then, the pursuer will reach the unsafe zone of the evader in at most ten seconds. Therefore, to avoid the collision, the action taken by evader should move the reachable set out of the position where the pursuer is.

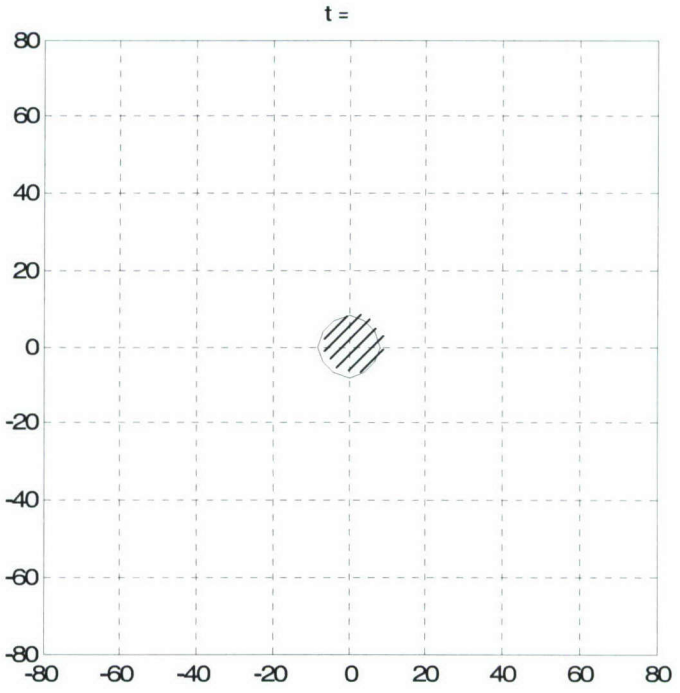

Figure 3. Unsafe zone for evader at t=0*s*

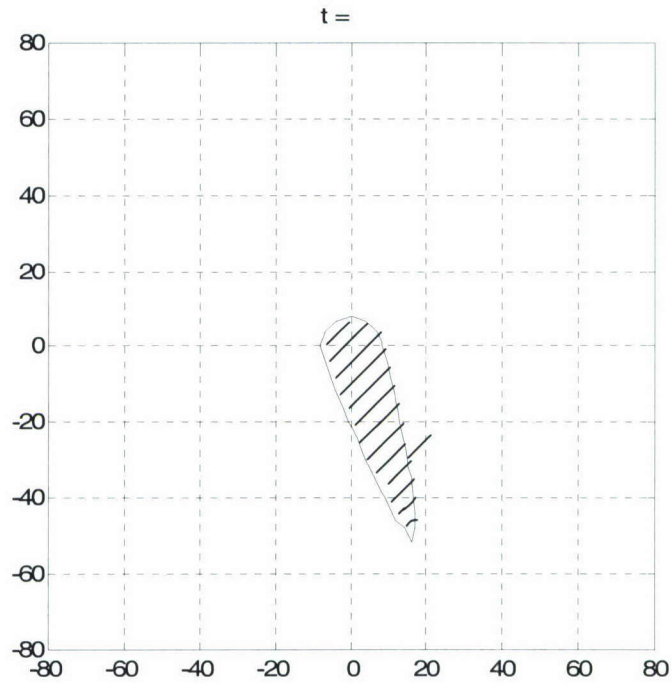

Figure 4. Backward reachable set for evader at t=-10s

In Figure 5, we keep the envelop of the backward reachable set for $\psi_r = \frac{\pi}{3}$ (illustrated in bold dashed line), and add another envelop for the backward reachable set at $\psi_r = \frac{\pi}{6}$ (illustrated in solid line). In the above example, if the evader detects the pursuer is at position A in the Figure 5, which is unsafe for the evader if it does not take any action. The two objects will collide after ten seconds because that the A point is within the ten seconds backward reachable set at $\psi_r = \frac{\pi}{3}$. The ten seconds backward reachable set for $\psi_r = \frac{\pi}{6}$ is shown in solid line in Figure 5. The point A is out of this set. Also, we notice that two reachable sets are partially overlapped which is marked by the cross-line shadow in Figure 5. The above information indicates that the shadowed area without cross-line is unsafe for $\psi_r = \frac{\pi}{3}$, but is safe for $\psi_r = \frac{\pi}{6}$, therefore, when the pursuer enters the shadowed area at point A, the evader predicts that the pursuer

enters the unsafe zone with $\psi_r = \frac{\pi}{3}$ after ten seconds, then the evader immediately adjusts its orbit by casting a proper Δv to the original velocity to change the angle ψ_r such that $\psi_r = \frac{\pi}{6}$. Because that the A point is not in the ten seconds backward reachable set for $\psi_r = \frac{\pi}{6}$, it will not enter the unsafe zone of the evader after ten seconds, hence the collision is avoided. It should be noted, because two reachable sets are partially overlapped, if the pursuer is in the overlapped area of both sets, then whatever the $\psi_r = \frac{\pi}{3}$ or $\psi_r = \frac{\pi}{6}$, the collision cannot be avoided. To avoid the collision, a different Δv, which can move the backward reachable set of the evader out of the current position of the pursuer, is necessary to be added to the evader velocity. This example demonstrates how we can use the backward reachability property to perform the collision avoidance maneuver between the evader and the pursuer.

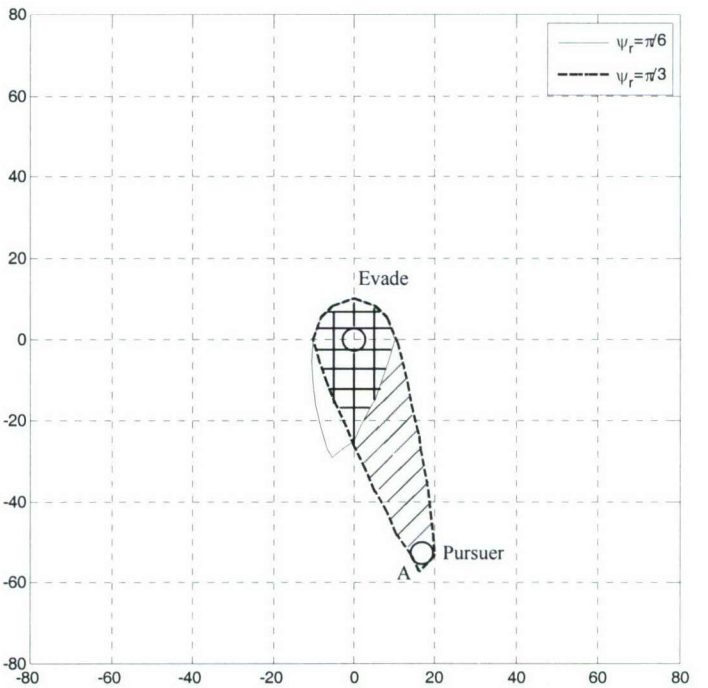

Figure 5. Evader changes the velocity to avoid the potential collision

Stability of a Hybrid System

To study the stability property of the hybrid system, we first define the equilibrium point of a hybrid system.

Definition 1 (Equilibrium Point): For a hybrid system *H*, a point $x = 0$ is an equilibrium point of *H* if and only if it satisfies the following conditions.

1. If $x = 0 \notin G(q,q')$, then $f(q,x,a,b,\delta_a,\delta_b) = 0$ for all $q \in Q$ when $a = b = \delta_a = \delta_b = 0$;
2. If $x = 0 \in G(q,q')$, then $R(q,q',0) = \{0\}$.

Here we note that the equilibrium point is not necessarily at $x = 0$, it can be any point and we use the same method to analyze the stability property of the equilibrium point.

Then, the definition of stability of an equilibrium point is described as follows.

Definition 2 (Stable Equilibrium): Assume $x = 0$ is an equilibrium point of the hybrid system *H*, then, $x = 0$ is stable if for any $\varepsilon > 0$, there exists a $\delta > 0$ such that for any execution $\zeta_H(\tau,q,x,t,a,b,\delta_a,\delta_b)$ that starts from (q_0,x_0), if $\|x_0\| < \delta$, $\|x(t)\| < \varepsilon$ for all $t \in \tau$.

Definition 3 (Asymptotically Stable Equilibrium): Assume $x = 0$ is an equilibrium point of the hybrid system *H*, then, $x = 0$ is asymptotically stable if there exists a $\delta > 0$ such that for any execution $\zeta_H(\tau,q,x,t,a,b,\delta_a,\delta_b)$ that starts from (q_0,x_0), if $\|x_0\| < \delta$, $\lim_{t\to\tau_\infty}\|x(t)\| = 0$.

According to Branicky [5], the stability property can be determined using multiple Lyapunov functions. For our hybrid system model which is defined at the beginning of the paper, we have the following theorem.

Theorem: Consider a hybrid system *H* with the equilibrium *x*=0, $|Q| < \infty$, and $R(q,q',x) = \{x\}$. For each *q*, we consider an open set $D_q \subseteq X$, such that $x = 0 \in D_q$. If

there exists a function V_q, which is continuously differentiable in x, satisfies the following conditions:

1. $V_q(q,0)=0$,
2. $V_q(q,x)>0$ for all $x \in D_q \setminus \{0\}$,
3. $\frac{dV_q}{dx}(q,x) f(q,x) \le 0$ for all $x \in D_q$.

And if for all $(\tau, q, x, a, b, \delta_a, \delta_b) \in H$, and all $q' \in Q$, if $q(\tau_i) = q'$, then $V_{q'}(q',x) < V_q(q,x)$, in other words, the sequence $\{V_q(q(\tau_i), x(\tau_i)) : q(\tau_i) = q'\}$ is non-increasing, then x=0 is a stable equilibrium of H.

References

[1] I. Mitchell, A. Bayen, and C. J. Tomlin. Validating a Hamilton-Jacobi approximation to hybrid system reachable sets. In M. D. Di Benedetto and A. Sangiovanni-Vincentelli, editors, Hybrid Systems: Computation and Control, number 2034 in Lecture Notes in Computer Science, pages 418-432. Springer Verlag, 2001.

[2] Ian Mitchell. *Games of two identical vehicles*. Technical Report SUDAAR 740, Department of Aeronautics and Astronautics, Stanford University, Stanford, CA, July 2001.

[3] Claire J. Tomlin. *Hybrid Control of Air Traffic Management Systems*. PhD thesis, Department of Electrical Engineering and Computer Science, University of California, Berkeley, 1998.

[4] Ian Mitchell. *Application of Level Set Methods to Control and Reachability Problems in Continuous and Hybrid Systems*. PhD thesis, Stanford University, California, August 2002

[5] MS Branicky, *Multiple Lyapunov functions and other analysis tools for switched and hybrid systems*. Automatic Control, IEEE Transactions on, 1998